Claims

1. A computer system (900) comprising:
a source file repository (100) storing a plurality of active source files (AS1, AS2, AS3) belonging to a component (C1); and
a central compilation service (200) that, upon receiving (410) an activation request for at least one inactive source file (IS1) of the component (C1), compiles (420) the component (C1) using the at least one inactive source file (IS1) and, in case the compilation is successfully completed, initiates (430) a transfer (440) of the at least one inactive source file (IS1) to the plurality (P1) of active source files.

2. The computer system of claim 1, wherein the transfer (440) each inactive source file (IS1) depends on a change mode selected from the group of:
adding the inactive source file (IS1) to the plurality (P1) of active source files in case the plurality has no corresponding active source file;
replacing a corresponding active source file (AS1) with the inactive source file (IS1) in case the corresponding active source file (AS1) is outdated; and
deleting the corresponding active source file (AS1) from the plurality (P1) of active source files in case the inactive source (IS1) file is deleted.

3. The computer system of claim 1 or 2, further comprising:
a runtime archive storage (300) to store (450) a compilation result (CR1) of the component (C1) in case the compilation of the component (C1) is successfully completed.

4. The computer system of any one of the claims 1 to 3, wherein the at least one inactive source file (IS1) is stored in a further source file repository (100').

5. The computer system of any one of the claims 1 to 4, wherein the central compilation service (200) notifies (470) a changer (90) of the inactive source file (IS1) in case the compilation of the component fails (460).

6. The computer system of any one of the claims 1 to 5, wherein the central compilation service assigns a component status to the component depending on the result of the compilation, the component status being selected from the group of:
ready in case the compilation of the component is successfully completed; and
broken in case the compilation of the component fails.

7. The computer system of any one of the claims 1 to 6, wherein the central compilation service assigns a component status dirty to a further component that depends on the component in case the compilation of the component is successfully completed.

8. The computer system of any one of the claims 1 to 7, wherein the central compilation service (200) performs an incremental build when compiling (420) the component (C1) having a dependency (D2) on a further component (C2), the incremental build using a component dependency evaluator (210) to determine the dependency (D2) and to provide a previously obtained compilation result of the further component (CR2), and to provide the at least one inactive source file (IS1) and the plurality (P1) active source files of the component (C1).

9. The computer system of any one of the claims 1 to 8, wherein the central compilation service (200) performs a parallel build when compiling the component (C1) by evaluating dependencies of the component (C1) on further components and compiling the component (C1) and at least one of the further components in parallel based on the dependencies.

10. The computer system of claim 9, wherein the parallel build is performed by a cluster of build computers.

11. A central compilation service (200) comprising instructions that when loaded into a memory of a computer system (900) and executed by at least one processor of the computer system (900), perform the steps:

receiving (410) an activation request for at least one inactive source file (IS1) of the component (C1);

compiling (420) the component (C1) using the at least one inactive source file (IS1);

in case the compilation is successfully completed, initiating (430) a transfer (440) of the at least one inactive source file (IS1) to a plurality (P1) of active source files; and

in case the compilation fails (460), notifying (470) a changer (90) of the inactive source file (IS1).

12. The central compilation service (200) of claim 11, wherein the compiling step (420) is performed as an incremental build, the component (C1) having a dependency (D2) on a further component (C2), the incremental build using a component dependency evaluator (210) to determine the dependency (D2) and to provide a previously obtained compilation result of the further component (CR2), and to provide the at least one inactive source file (IS1) and the plurality (P1) active source files of the component (C1).

13. The central compilation service (200) of any one of the claims 11 to 12, wherein the compiling step (420) is performed as a parallel build by evaluating dependencies of the component (C1) on further components and compiling the component (C1) and at least one of the further components in parallel based on the dependencies.

14. The central compilation service (200) of claim 13, wherein the parallel build is performed by a cluster of build computers.

15. A central compilation computer (903) comprising a central compilation service (200) according to any one of the claims 11 to 14.

16. A source file activation method comprising the steps:

receiving (410) at a central compilation service (200) an activation request for at least one inactive source file (IS1) of a component (C1);

compiling (420) the component (C1) using the at least one inactive source file (IS1);

in case the compilation is successfully completed, initiating (430) a transfer (440) of the at least one inactive source file (IS1) to a plurality (P1) of active source files; and

in case the compilation fails (460), notifying (470) a changer (90) of the inactive source file (IS1).

17. The method of claim 16, wherein the transfer (440) of each inactive source file (IS1) depends on a change mode selected from the group of:

adding the inactive source file (IS1) to the plurality (P1) of active source files in case the plurality has no corresponding active source file;

replacing an active source file (AS1) with the inactive source file (IS1) in case the corresponding active source file (AS1) is outdated; and

deleting the corresponding active source file (AS1) from the plurality (P1) of active source files in case the inactive source (IS1) file is deleted.

18. The method of claim 16 or 17, further comprising the step:

storing (450) a compilation result (CR1) of the component (C1) in a runtime archive storage (300) in case the compilation of the component (C1) is successfully completed.

19. The method of any one of the claims 16 to 18, further comprising the steps:

assigning a component status to the component (C1) depending on the result of the compilation, the component status being selected from the group of: ready in case the compilation of the component is successfully completed, and broken in case the compilation of the component fails; and

assigning a component status dirty to a further component in case the further component depends on the component and the compilation of the component is successfully completed.

20. The method of any one of the claims 16 to 19, wherein the compiling step (420) is performed as an incremental build, the component (C1) having a dependency (D2) on a further component (C2), the incremental build using a component dependency evaluator (210) to determine the dependency (D2) and to provide a previously obtained compilation result of the further component (CR2), and to provide the at least one inactive source file (IS1) and the plurality (P1) active source files of the component (C1).

21. The method of any one of the claims 16 to 20, wherein the compiling step (420) is performed as a parallel build by evaluating dependencies of the component (C1) on further components and compiling the component (C1) and at least one of the further components in parallel based on the dependencies.

22. The method of claim 21, wherein the parallel build is performed by a cluster of build computers.

23. A method for validating software comprising the steps:

a local file system (110) retrieving (1a) a source file of a component referencing a referenced component from a source file repository of a source control system (902);

the local file system (110) obtaining (1b) a compilation result of the referenced component from a runtime archive storage (RTA) (300);

an integrated development environment (IDE) (110) receiving a modification (EDIT) of the source file;

upon having received the modification, the IDE (110) transferring (2) the source file to a local build tool (140);

the local build tool (140) retrieving (3) the compilation result from the local file system (110);

the local build tool (140) locally compiling (4) the component that includes the modified source file by using the compilation result of the referenced component resulting in a new compilation result of the component;

storing (5) the new compilation result in the local file system (110);

checking in (7) the modified source file into the source file repository, the modified source file becoming an inactive source file of the source file repository;

launching (8) an activation request with regards to the inactive source file directed to a central compilation service (CCS) (200);

the CCS (200) loading (9a) the inactive source file and corresponding active source files of the component from the source file repository, and retrieving (9b) the compilation result of the referenced component from the RTA 300;

the CCS 200 centrally compiling (10) the component; and

in case of successful central compilation (10) the CCS (200) triggering (12) the activation of the successfully compiled inactive source file in the source file repository.

24. The method of claim 23, comprising the further step:

the IDE (120) deploying (6) the new compilation result to a local runtime (130) for test purposes prior to the checking in step (7).

25. The method of claims 23 or 24, comprising the further step:

the CCS (200) making available (11) the result of the central compilation in the RTA (300).

26. The method of any one of the claims 23 to 25, comprising the further step:

the CCS (200) storing an error result in case an error occurs during the central compilation (10), and making available the error result to a changer (90) of the inactive source file causing the error.

27. A local development computer (901) configured to execute the steps retrieving (1a), obtaining (1b), transferring (2), retrieving compilation result (3), locally compiling (4), storing (5), deploying (6) and launching (8) of the method according to the claims 23 or 24.